81

PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : G06F 17/50

A1

(11) International Publication Number: WO 99/28840

(43) International Publication Date: 10 June 1999 (10.06.99)

(21) International Application Number: PCT/US98/24909

(22) International Filing Date: 24 November 1998 (24.11.98)

(30) Priority Data:
08/980,631 1 December 1997 (01.12.97) US

(71) Applicant: IMPROV SYSTEMS, INC. [US/US]; Suite 407L, 100 Cummings Center, Beverly, MA 01915 (US).

(72) Inventors: USSERY, Cary; 27 Elm Street, Hamilton, MA 01982 (US). LEVIA, Oz; 1622 Hummingbird Lane, Sunnyvale, CA 94087 (US). RYAN, Raymond; 953 Mt. Carmel Drive, San Jose, CA 95120 (US).

(74) Agent: RAUSCHENBACH, Kurt; Testa, Hurwitz & Thibeault, LLP, High Street Tower, 125 High Street, Boston, MA 02110 (US).

(81) Designated States: JP, KR, European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published

With international search report.
Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.

(54) Title: METHOD OF GENERATING APPLICATION SPECIFIC INTEGRATED CIRCUITS USING A PROGRAMMABLE HARDWARE ARCHITECTURE

(57) Abstract

A method for generating an application specific integrated circuit including providing a software configurable semiconductor integrated circuit having a fixed hardware architecture that includes a plurality of task engines. A high-level language compiler is provided that compiles a user created high-level language program that defines the application specific integrated circuit. The compiler parses the program into a plurality of microtasks for instructing the plurality of task engines to implement the application specific integrated circuit.

Method of Generating Application Specific Integrated Circuits Using a Programmable Hardware Architecture

Field of the Invention

The invention relates generally to the field of application specific integrated circuits. In particular, the invention relates to methods of generating application specific integrated circuits using a fixed but configurable hardware architecture.

Background of the Invention

Custom integrated circuits are widely used today in the electronics industry. The demand for custom integrated circuits is rapidly increasing because of a dramatic growth in the demand for highly specific consumer electronics and a trend towards increased product functionality. Also, the use of custom integrated circuits is advantageous because they reduce system complexity and, therefore, lower manufacturing costs, increase reliability and increase system performance.

There are numerous types of custom integrated circuits. One type is programmable logic devices (PLDs) including field programmable gate arrays (FPGAs). FPGAs are designed to be programmed by the end user using special-purpose equipment.. Programmable logic devices are, however, undesirable for many applications because they operate at relatively slow speeds, have relatively low capacity, and have relatively high cost per chip.

Another type of custom integrated circuit is application-specific integrated circuits (ASICs) including gate-array based and cell-based ASICs which are often referred to as "semicustom" ASICs. Semicustom ASICs are programmed by defining either a) defining the placement and interconnection of a collection of predefined logic cells which are used to create a mask for manufacturing the IC (cell-based) or b) defining the final metal interconnection layers to lay over a predefined pattern of transistors on the silicon (gate-array-based). Semicustom ASICs can achieve high performance and high integration but can be undesirable because they have relatively high design costs, have relatively long design cycles (time it take to transform given

consuming and expensive iterations. Customer modifications or problems occurring during the design cycle may require costly redesign and long delays.

Because of the trend towards increased product functionality in the electronic industry, the complexity of custom integrated circuits is rapidly increasing. The level of skill required to generate custom integrated circuits and the design cycle time is also rapidly increasing. Consequently, prior art methods of generating custom integrated circuits are becoming increasingly inadequate. There currently exists a need for a method of generating application specific integrated circuits that reduces the design cycle time of custom integrated circuits. There also exists a need for a method of generating application specific integrated circuits that allows for modification during the design cycle.

Summary of the Invention

It is therefore a principal object of this invention to greatly reduce the number of steps that it takes to produce an application specific integrated circuit and, therefore, to greatly reduce the design cycle time and the manufacturing cost. It is another object of this invention to provide a method of generating an application specific integrated circuit that easily implements design modifications during the design cycle.

It is another object of this invention to reduce the engineering skill level required to create an application specific integrated circuit. A principal discovery of the present invention is that a custom integrated circuit can be produced by programming a fixed architecture integrated circuit using a high-level object oriented programming language. It is another principal discovery that a custom integrated circuit can be produced by as little as two steps comprising describing the desired functionality of the integrated circuit in an object oriented programming language and compiling the object oriented program onto the fixed programmable architecture.

It is yet another principal discovery that a compiler can be used to perform high level synthesis to map specific functions of an application onto task engines comprised of set data paths in the PSA IC thereby eliminating time intensive analysis of possible data path.

Accordingly, the present invention features a method for generating an application specific integrated circuit that includes providing a software configurable semiconductor integrated circuit

engine. At least one task engine may be included that programs an input and an output interface for accepting data with a communication protocol. The hardware architecture may also include a program memory associated with each of the task engines for storing microtasks for instructing the task engines.

A high-level language compiler compiles a user created high-level language program that defines the application specific integrated circuit. The compiler parses the program into a plurality of microtasks that instruct the plurality of task engines to implement the application specific integrated circuit. The apparatus may also include a software simulator that evaluates particular architectures.

The component libraries define a set of component objects that are used as building blocks in developing an application for the PSA IC. The component objects are class definitions derived from a base class. Each component object has predefined input and output channels. The component objects may include code for compression, coding, digital audio/video, connectivity, and switching functions. For example, component objects may include code for a JPEG codec, an MPEG codec, an ATM switch, or a USB bus slave.

The application framework libraries define a complete description of a system that can be run on a PSA IC. Examples of such systems are code multimedia, video conferencing, set top box and audio systems. Specifically, an application framework library may include code for image processing, user control and image compression functions of a digital camera.

Using application libraries 58 reduces the design cycle time and the level of skill required to produce the custom integrated circuit. Using the application libraries also improves system performance because the libraries are optimized for particular target task engines. In addition, using application libraries makes the system development more intuitive because users can design a system from high level blocks.

The user program 56 containing the custom code that defines the system specification for the ASIC and the application libraries 58 is parsed by the PSA Compiler 52. The PSA compiler 52 converts the user program 56 into a program image 60 of the system specifications for the ASIC that comprises a series of microtasks. Each microtask is a Very Long Instruction Word (VLIW) program for a target task engine in the PSA IC 54.

The PSA analysis system 62 is a graphical analysis environment that allows the user to analyze certain characteristics of their application running on a target PSA IC. It includes a PSA simulator, PSA configuration tool, and graphical user interface (GUI) environment. The PSA simulator is a software model of a specific configuration of the PSA IC hardware that can execute the program image 60 produced by the PSA compiler 52. The PSA configuration tool allows the user to select between different configurations of the PSA IC. The configuration information is used by the PSA compiler 52 and by the simulator to accurately reflect the characteristics of the target PSA IC 54. The graphical user environment allows the user to interact with the different analysis tools through a graphical windowing interface.

global analyzer 106 allocates memory segments for communicating in shared memory and for local storage required by the task in private memory. The global analyzer 106 also allocates memory locations based on data lifetime analyses.

The global analyzer 106 also performs datapath analysis. The global analyzer 106 determines the optimal task engines on which to execute the functionality of a given task. This is done by analyzing the required operations of the task (addition, multiplication, etc.) and matching those to a task engine that can optimally support the collection of operations required by the task. For example, if a task includes a number of multiplication operations, the global analyzer 106 maps the task onto a task engine that includes a multiplier as one of its computation units.

The global analyzer 106 also performs task scheduling. The global analyzer 106 determines the relationships between tasks and inserts code to efficiently order the execution of the tasks so as to optimize the control flow and data relationships between tasks. In addition, the global analyzer 106 performs system level optimization and allocation. That is, the global analyzer 106 processes the entire program (rather than individual modules) and makes global decisions and optimizations for program execution and memory references.

The global analyzer 106 also inserts direct memory references, rather than memory addressing, in the instruction set. Direct memory addresses can be used because the PSA IC hardware architecture implements the overall system including the memory used for data storage. Direct memory addressing is advantageous because it minimizes the number of required instructions and enables asymmetric pipelining of instructions. Finally, the global analyzer 106 generates intermediate forms that are data structures which define a network of tasks where each task is represented by a control/dataflow graph.

A task analyzer and code generator 108 processes the tasks allocated to specific task engines and generates a series of microtask definitions that run on particular task engines. The microtasks are atomic execution units that are non-preemptive. That is, once they are executed, they will run to completion without being interrupted. In one embodiment, the task analyzer and code generator 108 uses program decomposition techniques to decompose tasks into threads and then, through data flow analysis, decompose the threads into microtasks. The task analyzer and

152 with predefined communication mechanisms and memory structures. The task engines 152 are high performance data paths that are programmed with a Very Long Instruction Word (VLIW). Programming with VLIWs allows the compiler to select the optimum task engine to perform some part of the ASIC operation.

The task engines 152 are optimized for different types of tasks. Some tasks will require advanced processing capabilities and, therefore, will require ALUs and multipliers in the task engine's datapath. Other tasks may simply be used to transform sequences of data and, therefore, will require only a simple datapath consisting of little more than a shift register.

The PSA hardware 150 can communicate with a diverse range of other circuits through various protocols. The inputs and outputs of the PSA hardware 150 are programmed using special input/output task engines 154 (I/O task engines) that interface with input/output memory (I/O memory) 156. The I/O memory 156 is random-access memory (RAM) that stores data that is input from and output to external pin modules (not shown) of input/output sections 160 (I/O sections). In one embodiment, there are at least two I/O memories 156 interfacing with at least two I/O sections 156.

In one embodiment, the external pin modules are configurable through pin configuration registers. The I/O task engines 154 associated with the I/O memory 156 and the I/O section 160 can be programmed to write new values into these configuration registers in order to configure the operation of the pin module.

In one embodiment, the pin-to-memory-location mapping of the PSA hardware 150 is predefined (i.e., not programmable). The predefined mapping allocates one or more pins into a word memory location. Pins may be bundled so that, for example, 16 pin values may be mapped into a single word. This enables programs to allocate sets of pins to commonly bundled values (e.g., data busses).

The task engines 152, including the I/O sections 160, communicate by the shared memory modules 158. The shared memory modules 158 are typically random-access memories (RAM) that can be written to and read from by the task engines. Communication between tasks is performed by having one task write to and another task read from the shared memory 158.

bus 210 allows the task engine 200 to receive tasks from the scheduler and other task engines connected to the queue bus 210.

A task controller 212 that is in communication with the task queue 208 and the data bus 206 manages the execution of microtasks that are stored on the task queue 208. The task controller 212 includes a program counter (not shown) that is updated to include the next microtask to run and manages the loading and execution of instructions for that microtask.

Instruction memory 214 is in communication with the task controller 200. The instruction streams for each microtask assigned to run on particular task engine 206 are loaded into the instruction memory for the particular task engine. In one embodiment, the instruction memory 214 is non-volatile memory such as flash memory.

An address generation unit 216 takes memory references from the task controller 212 and converts them to actual memory locations in either the private data memory 202 or in shared memories 203. There are two advantages to this approach: shorter immediate addressing saves program memory and vector address generation increases performance and saves execution resources. This causes the data in those memory locations to be made available to the appropriate data bus.

Different task engines are designed to have different configurations and types of computation units in communication with the data bus 206. A computation unit is an on-chip block which performs some computational function. Typical examples of computation units are multipliers 218, data processing units (DPUs) 220, and shifters units 222.

What is claimed is:

1. A method for generating an application specific integrated circuit, the method comprising:

   a) providing a software configurable semiconductor integrated circuit having a fixed hardware architecture that includes a plurality of task engines and configurable I/O;

   b) providing a high-level language compiler; and

   c) compiling a user created high-level language program with the compiler that defines the application specific integrated circuit, the compiler parsing the program into a plurality of microtasks for instructing the plurality of task engines to implement the application specific integrated circuit.

2. The method of claim 1 further comprising providing the user created high-level language program that defines the application specific integrated circuit.

3. The method of claim 1 wherein the high-level language is an object oriented or component-based programming language.

4. The method of claim 3 further comprising providing at least one object oriented class library that is compilable with the high-level language compiler to generate microtasks for instructing task engines to perform algorithms, data communications or data manipulation.

5. The method of claim 3 wherein the object oriented programming language is a Java programming language.

6. The method of claim 1 wherein each of the microtasks comprises a Very Long Instruction Word program that instructs a task engine.

7. The method of claim 1 further comprising loading the microtasks into program memory associated with the task engines.

8. The method of claim 1 wherein the compiler is optimized to select an optimum task engine for each of the microtasks.

17. The apparatus of claim 13 wherein the fixed hardware architecture further comprises a program memory associated with each of the task engines for storing microtasks for instructing the task engines.

18. The apparatus of claim 13 wherein each of the plurality of task engines is programmable with a unique Very Long Instruction Word instruction set.

19. A method for generating an application specific integrated circuit, the method comprising:

a) providing a software configurable semiconductor integrated circuit having a fixed hardware architecture including a plurality of task engines;

b) providing a high-level language compiler;

c) providing a user created high-level language program for the high-level language compiler that defines the application specific integrated circuit; and

d) compiling the program with the compiler, the compiler parsing the program into a plurality of microtasks for instructing the plurality of task engines to implement the application specific integrated circuit.

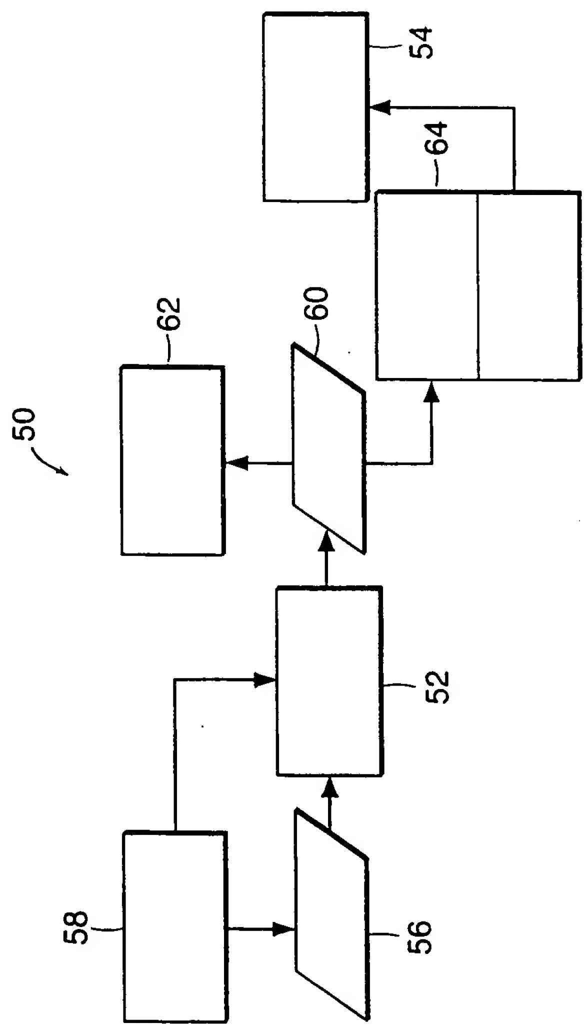

FIG. 2

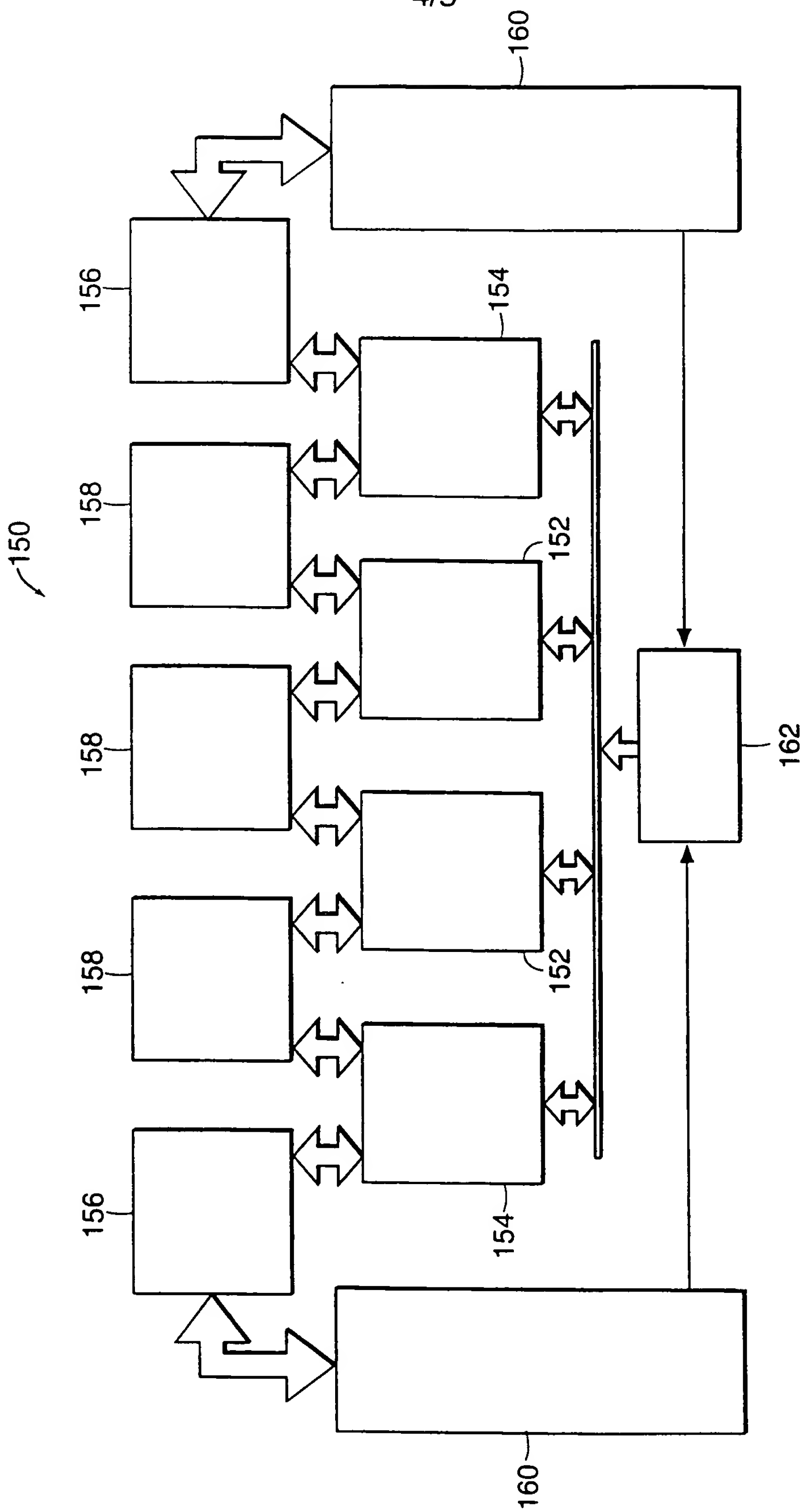

FIG. 4

INTERNATIONAL SEARCH REPORT

International Application No
PCT/US 98/24909

A. CLASSIFICATION OF SUBJECT MATTER
IPC 6 G06F17/50

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 6 G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X A	US 5 524 244 A (KRASSOWSKI ANDREW J ET AL) 4 June 1996 see abstract see column 6, line 1 - line 34 see column 50, line 35 - column 52, line 43 see column 59, line 10 - column 60, line 32 see column 72, line 30 - column 74, line 56 see figures 1,3,10,12 --- -/--	1,2,13, 19 7,8,10, 16,17

[X] Further documents are listed in the continuation of box C.

[X] Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
15 April 1999	22/04/1999

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Guingale, A

2

INTERNATIONAL SEARCH REPORT

Information on patent family members

International Application No
PCT/US 98/24909

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 5524244 A	04-06-1996	US 5287511 A	15-02-1994
		US 5068823 A	26-11-1991
		AU 2902192 A	21-05-1993
		CA 2120232 A	29-04-1993
		EP 0681718 A	15-11-1995
		IL 103433 A	18-03-1997
		JP 7504281 T	11-05-1995
		WO 9308524 A	29-04-1993
		AU 3974289 A	19-02-1990
		CA 1337531 A	07-11-1995
		US 5263143 A	16-11-1993
		WO 9001214 A	08-02-1994
		US 5590349 A	31-12-1996
		AU 665927 B	25-01-1996
		AU 7980391 A	10-12-1991
		CA 2084420 A	19-11-1991
		EP 0530310 A	10-03-1993
		IL 98161 A	10-01-1997
		IL 119396 A	20-11-1997
		JP 5509425 T	22-12-1993
		US 5428749 A	27-06-1995
		WO 9118342 A	28-11-1991
		AU 8726791 A	15-04-1992
		WO 9205496 A	02-04-1992
		AU 1753792 A	06-10-1992
		WO 9215949 A	17-09-1992
WO 9410627 A	11-05-1994	US 5535342 A	09-07-1996
		AU 5458194 A	24-05-1994
		CA 2148813 A	11-05-1994
		EP 0746812 A	11-12-1996
		JP 8504285 T	07-05-1996
		AU 5593594 A	24-05-1994
		CA 2148814 A	11-05-1994
		EP 0667010 A	16-08-1995
		JP 8504514 T	14-05-1996
		WO 9410624 A	11-05-1994
		US 5497498 A	05-03-1996
		US 5603043 A	11-02-1997
		US 5857109 A	05-01-1999